

# Cosmic-ray Shower Library (CRY)

Cosmic-ray Physics Team  
Lawrence Livermore National Laboratory

August 20, 2007

## 1 Introduction

The CRY software library generates correlated cosmic-ray particle shower distributions at one of three elevations (sea level, 2100m, and 11300m) for use as input to transport and detector simulation codes. Particles from cosmic ray showers [1] over a wide range of energies (1 GeV – 100 TeV primary particles and 1 MeV – 100 TeV secondary particles) are generated from data tables. These tables are derived from full MCNPX [2] simulation for muons, neutrons, protons, electrons, photons, and pions for several altitudes. The CRY software package generates shower multiplicity within a specified area (up to 300m by 300m) as well as the time of arrival and zenith angle of the secondary particles. For a more detailed discussion of the cosmic-ray model used to generate the input tables see [3]. The description of options and the basic C++ interface are given below. The software library and examples can be downloaded from <http://nuclear.llnl.gov/CNP/simulation>.

## 2 Creating the library and test programs

Type 'make' from the top-level directory of the software distribution. This will create the software library `lib/libCRY.a` and multiple test executables in the `test` directory. There are three test programs included in the distribution: "testMain.cc", "testAlts.cc", and "testRoot.cc", the latter requires the "root" package (<http://root.cern.ch>) to create summary histograms of the results and is not generated by default. The execution examples for each program are:

```
./testMain setup.file [nevents]
./testAlts
./testRoot setup.file [nevents]
```

The programs must be run from the `test` directory. The number of events to run is an optional second argument for `testMain` and `testRoot`. Reference output files (`.ref`) are included in the software distribution.

There are also examples of using CRY as the input source for the Geant4 and COG Monte Carlo transport codes in the 'geant' and 'cog' directories, respectively.

### 3 Available options

Options can be specified in setup.file (for the test programs). In the interface to Geant4 and COG the options can also be specified in the input deck. The option syntax is always <token> <value> pairs where white space is ignored.

#### Types of secondary particles to be returned

By default all particle types are returned. Each can be disabled independently using these flags (n=1 return the particle, 0 do not return the particle)

```
returnNeutrons n
returnProtons n
returnGammas n
returnElectrons n
returnMuons n
returnPions n
```

#### Altitude (meters)

```
altitude n
```

Allowed values are 0, 2300, and 11300.

#### Latitude (degrees)

```
latitude n
```

#### Date

```
date n
```

Controls the location in solar sun spot cycle.

#### Control of the # of particles

```
nParticlesMin n
```

```
nParticlesMax n
```

Showers with more than the specified value (n) are truncated.

#### The lateral size of interest (meters)

```
subboxLength n
```

Particles are returned inside a box of n by n. The maximum allowed value is 300m. Data tables are provided for 1, 3, 10, 30, 100, and 300m. For box sizes in between these discrete values, the next largest table will be utilized and particles outside of the specified window will be dropped.

## 4 Interface

To specify the location of the input data and the user selected setup:

```
CRYSetup *setup=new CRYSetup(setupString,"./data");  
CRYGenerator gen(setup);
```

The first argument to CRYSetup is a string containing the contents of the setup file, and the second is the location of the data tables (in src/data).

To generate all particles of interest for a single shower use:

```
std::vector<CRYParticle*> *ev=new std::vector<CRYParticle*>;  
ev->clear();  
gen.genEvent(ev);
```

The shower is returned as a list of particles, represented by CRYParticle. The particle type, energy, location, timing, and direction cosines are accessible (see CRYParticle.h for more information). The default units are:

```
Energy=MeV  
Position=meter,  
Time=second.
```

### Elapsed time

The generator can also be queried for the elapsed time in the simulation for normalization purposes via `gen.timeSimulated()`.

### Random number generator

The random number generator can be changed by passing a pointer to the location of the function that returns random numbers:

```
CRYSetup::setRandomFunction(double (*newFunc)(void))
```

## References

- [1] "An estimate of the secondary-proton spectrum at small atmospheric depths," P. Papini, C. Grimani and S.A. Stephens, *Nuovo Cimento* 19C, 367 (1996).
- [2] MCNPX User's Manual, Version 2.3.0, Laurie Waters, ed., LA-UR-02-2607 (2002).
- [3] "Monte Carlo Simulation of Proton-induced Cosmic-ray Cascades in the Atmosphere," Lawrence Livermore National Laboratory, UCRL-TR (2007).